AD-A082 188   INSTITUTE FOR DEFENSE ANALYSES ARLINGTON VA PROGRAM --ETC F/G 15/7
IDAHEX VERSION 2. VOLUME III. PLAYER'S MANUAL.(U)
MAY 79 P OLSEN   DAHC15-73-C-0200
UNCLASSIFIED IDA-P-1266-VOL-3   IDA/HQ-77-19282   NL

1 OF 1
AD
A082188

END
DATE
FILMED
4-80
DTIC

IDA PAPER P-1266

IDAHEX VERSION 2

VOLUME III: Player's Manual

Paul Olsen

May 1979

Prepared for

Office of the Assistant Secretary of Defense
(Program Analysis & Evaluation)

IDA

INSTITUTE FOR DEFENSE ANALYSES
PROGRAM ANALYSIS DIVISION

IDA Log No. HQ 77-19282

The work reported in this document was conducted under Contract No. MDA903 79 C 0320 for the Department of Defense. The publication of this IDA Paper does not indicate endorsement by the Department of Defense nor should the contents be construed as reflecting the official position of that agency.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER FINAL
4. TITLE (and Subtitle) IDAHEX VERSION 2 Vol III: Player's Manual		5. TYPE OF REPORT & PERIOD COVERED FINAL 6. PERFORMING ORG. REPORT NUMBER IDA PAPER P-1266
7. AUTHOR(s) Paul Olsen		8. CONTRACT OR GRANT NUMBER(s) DAHC 15 73 C 0200
9. PERFORMING ORGANIZATION NAME AND ADDRESS Institute for Defense Analyses 400 Army-Navy Drive Arlington, VA 22202		10. PROGRAM ELEMENT PROJECT TASK AREA & WORK UNIT NUMBERS Task PA&E 106
11. CONTROLLING OFFICE NAME AND ADDRESS Office of the Assistant Secretary of Defense (Program Analysis & Evaluation) Pentagon, Washington, DC 20301		12. REPORT DATE May 1979 13. NUMBER OF PAGES 77
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Office of the Assistant Secretary of Defense (Program Analysis & Evaluation) Pentagon Washington, DC 20301		15. SECURITY CLASS (of this report) UNCLASSIFIED 15a. DECLASSIFICATION DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Land Warfare, Ground Combat, Simulation, Interactive Model, War Game, Computer-Assisted War Game, Ground Forces, Amphibious Landings, Airborne Operations, Maneuver		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) IDAHEX is an interactive computer model of two-sided conventional land warfare. It keeps the players informed of the situation and accepts their instructions to their forces. Units can move by land, sea, or air. A unit's movement rate is variable, depending upon its posture, the conditions of its movement, and the adequacy of transport. Attrition in engagements is assessed by a heterogeneous Lanchester square process. Indirect supporting fire and direct air support		

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Item 20 continued

can be played. There is a simplified representation of air attacks on lines of communication and of engineering activities, including bridge building and mine laying. Supplies consumption can be assessed. and logistics can be played. The model recognizes severed lines of retreat and lines of supply and imposes appropriate consequences. The documentation consists of three volumes: (1) A Guide for Potential Users: (2) Game Designer's Manual: (3) Player's Manual.

IDA PAPER P-1266

IDAHEX
VERSION 2
VOLUME III: Player's Manual

Paul Olsen

May 1979

INSTITUTE FOR DEFENSE ANALYSES
PROGRAM ANALYSIS DIVISION
400 Army-Navy Drive, Arlington, Virginia 22202

Contract No. DAHC 15 73 C 0200
Task PA&E 106

PREFACE

IDAHEX is a computerized model of land warfare at the theater level. Volume 1 outlines IDAHEX as a war game that realistically represents maenuver. Volume 2, the *Game Designer's Manual*, comprehensively describes the model and its basic input data, the "game design data". This volume, the *Player's Manual*, gives enough information for someone with a modest knowledge of land warfare to play an IDAHEX game, which may have been designed by someone else.

Comments and inquires are welcomed. They should be directed to the author (commercial telphone 202-697-0584, autovon 227-0584).

CONTENTS

FIGURES

TABLES

1. INTRODUCTION

IDAHEX is a model of warfare implemented as a computer program. Usually, no disti⌐ction is drawn between the model and the program; this manuaı refers to them both as "IDAHEX". IDAHEX can be regarded as a system that is used by the "game designer" to design a war game and then by the players to play it. There are two sides in the war, "Red" and "Blue", inducing a "Red player" role and a "Blue player" role. Although both roles may be assumed by a single person, IDAHEX is careful to distinguish them and to prevent one player from giving commands to the other's forces. Although IDAHEX can be played in batch processing mode, interactive use is far more convenient. IDAHEX can be played interactively with each player on a separate terminal or with both players sharing a single terminal.

The following sections except Section 4 form a self-contained outline of the model's structure; to understand the model fully, the player must read the *Game Designer's Manual* (Volume 2). Sections 4 and 6 explain how the players communicate with IDAHEX. Although this manual is directed toward the players, it sometimes identifies game design variables by name, in italics, so that the game designer can recognize them and answer players' questions about the values he has assigned to them.

2. THE ELEMENTS OF PLAY

This section explains how IDAHEX structures the area of war, the forces, and maneuver.

2.1 THE AREA OF WAR

The game board is termed the "area of war", the area in which the forces exist. It is partitioned into congruent, regular hexagons, as Figure 2.1 illustrates. The hexagons are termed "cells". The depth of any cell, the distance from one side to the side directly opposite it, is fixed by the game design variable *depth*. The cells are numbered for identification. A cell may be "inactive": in effect, it is excluded from the area of war, and no forces may enter it.

A cell's "environment" is the complex of physical conditions in the cell--including weather--that affect cross-country movement, ground combat, or vulnerability to air strikes. Examples: clear, hilly, muddy, built-up, fortified. The environment is treated as though uniform throughout the cell. Two adjacent cells may be linked by a road system, a rail system, or both. Road and rail links are typed to distinguish them according to their suitability for movement. Since there may be many roads or railroads leading from one cell directly to an adjacent cell, the road link type or rail link type should be interpreted as a general characterization of trafficability, of how easy it is to get directly from one cell to the other by road or rail. Two adjacent cells need not be linked by road or rail. In that case, going directly from one to the other requires cross-country movement, air movement, or possibly sea movement. Between two adjacent cells there may be a barrier. Barriers include rivers, ridges, and, in general, any obstacles that significantly affect land movement or attack from one cell to another. A given barrier may be subclassified as a movement barrier (an obstacle or series of obstacles that impede unopposed movement), an attack barrier (an obstacle or series of obstacles that impede attack), or both.

The game designer must give the players a map of the area of war that identifies the cells by their numbers and indicates

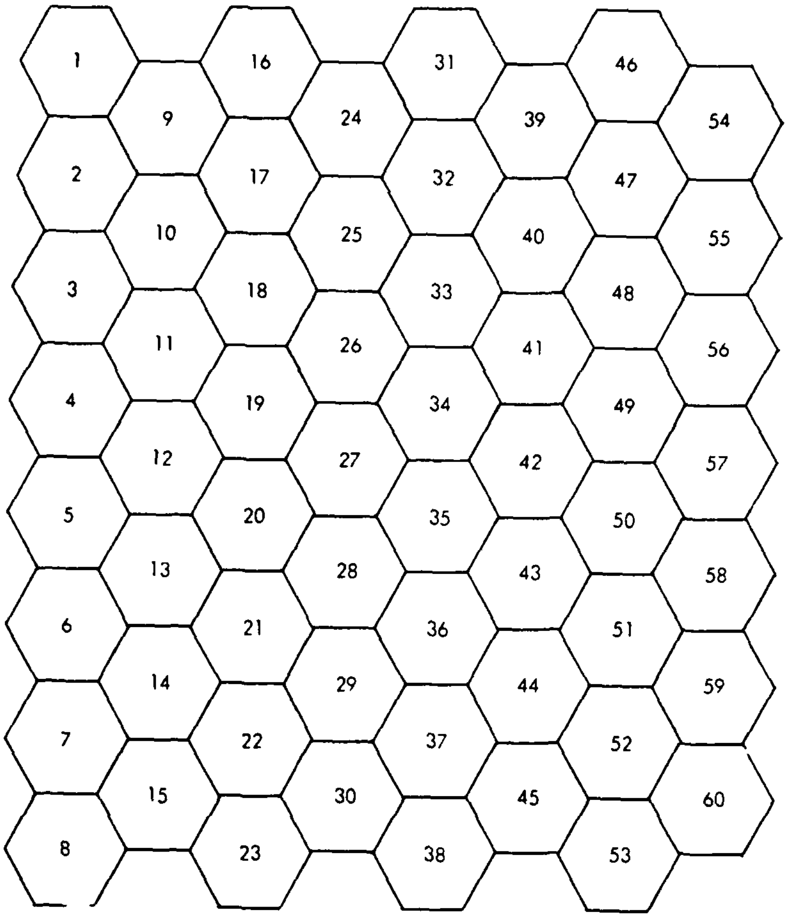

Figure 2.1. EXAMPLE OF AREA OF WAR

each cell's environment, the type of road link (if any) between each pair of adjacent active cells, the type of rail link (if any) between each pair of adjacent active cells, and the tyne of barrier between each pair of adjacent active cells. The cells' identification numbers never change but environment, road, rail, and barrier types may change during the game. IDAHEX informs the players of changes.

2.2 THE FORCES

There are two forces, Red and Blue. Each consists of indivisible "battle units", often called simply "units". The game designer assigns each unit a unique identification number, a positive integer. The identification number of every Red unit must be less than the identification number of every Blue unit. A unit's "type" determines the types of resources it is permitted to carry. A game's list of unit types might be:

1. Red motorized rifle division
2. Blue armored division
3. Red tank battalion
4. Red tank division
5. Red road transport unit
6. Blue rail transport unit
7. Blue infantry division
8. Blue road transport unit
9. Red rail transport unit

IDAHEX identifies unit types by numbers; therefore, the game designer should give the players a list like the one above.

2.2.1 Battle Unit Status

A battle unit's "status" is described by its location, posture, and objective.

Each battle unit is located in exactly one cell. The unit's location can not be fixed more precisely: there is no pretense of knowing, for example, that it is 3 km northeast of the cell's center. Its location *is* the cell. Several units may have the same location, even if they belong to opposite sides.

At any moment of the game, each battle unit is in one of six "posture classes":

-1. destroyed
 0. inactive
 1. hold

2. disengagement
3. movement
4. attack

A unit in posture class -1 or 0 is said to be "inactive". (Inversely, a unit in a positive posture class is said to be "active".) An inactive unit does not exist from the perspectives of other units. It can not move; it can not attack, nor can it be attacked. A unit in posture class -1 is a special kind of inactive unit: it was de-activated to represent its destruction, usually as a result of suffering intolerably high losses. A unit in posture class 0 is ordinarily a reinforcement or a package of replacements. It may become active (enter a positive posture class) later in the war. Its location is the cell where it is expected to enter the area of war if it becomes active, but while it remains inactive, it has no effect on enemy units passing through its location.

A unit in posture class 2 is trying to break contact with any enemy units it may be fighting, as the first step in changing location. Its "objective" is the cell toward which it is disengaging. A unit in posture class 3 is moving from its location to another cell, its objective. Ordinarily, a unit in posture class 4 is trying to enter a new location, which may or may not contain enemy units, but in some cases it is trying to revert from posture class 2, 3, or 4 to posture class 1 without changing location. In the former instance, its objective is the cell it seeks to enter; in the latter, its objective is just its present location.

Posture class 1 embraces all remaining activities as well as simple idleness. In particular, a unit in posture class 1 is not in the process of changing location. It may or may not be engaged. It objective is, by convention, its location.

Each positive posture class consists of from 1 to 10 postures. Posture class -1 consists of just one posture, numbered -10. The postures in posture class 0 are numbered 0 through 9, but IDAHEX does not distinguish one posture in posture class 0 from another. The postures in posture class 1 through 4 are numbered as follows:

10-19	hold
20-29	disengagement
30-39	movement
40-49	attack

These numbers are used to identify the postures in communications between the players and IDAHEX. Table 2.1 presents alternative ways of describing a unit's posture. The number of postures in a posture class and their interpretations depend upon the game design data. Therefore, the game designer must explain the

Table 2.1. EQUIVALENT DESCRIPTIONS OF POSTURE CLASS

in posture class 1;	in a hold posture;	holding
in posture class 2;	in a disengagement posture;	disengaging
in posture class 3;	in a movement posture;	moving
in posture class 4;	in an attack posture;	attacking

Table 2.2. ILLUSTRATIVE LIST OF POSTURES

Hold Postures

10. standard hold
11. halt, on roads
12. halt, dispersed off-road
13. delay
14. transfer
15. hasty defense

Disengagement Postures

20. standard disengagement
21. disengagement by road

Movement Postures

30. tactical march
31. administrative march (road march)
32. air movement

Attack Postures

40. standard attack
41. attack from administrative march
42. hasty attack

postures to the players, at least giving them a list like Table 2.2.

2.2.2 Resources

The game designer should give the players lists of the sides' resource types. For example, the list of Red resource types might be:

1. tanks
2. small arms and APCs
3. artillery
4. SAMs and AAA
5. trucks
6. ammunition
7. fuel and other consumables
8. tank crewmen
9. other personnel

The Blue list might be:

1. small arms and APCs
2. artillery
3. tanks
4. trucks
5. supplies
6. personnel

There is no correspondence between Red resource types and Blue resource types: in the example, Red type 3 resources are artillery while Blue type 3 resources are tanks, and Red has SAMs and AAA while Blue has none. The resource types must be listed in the following order:

ground-to-ground weapons
ground-to-air weapons
transport
supplies
personnel

These five resource categories combine to form larger categories:

materiel	ground-to-ground weapons	weapons	equipment
	ground-to-air weapons		
	transport		
	supplies		support
	personnel		

A category may include one or more resource types, or, with the exception of ground-to-ground weapons, it may be empty.

A player can learn what types of resources each side has and how the resource types are numbered by giving the command "display unit" (described in Section 4). Since diverse resources may be aggregated into a single type, the game designer probably should explain the method of aggregation--perhaps identifying a specific resource as the normative resource, or *numeraire*, of each type.

A battle unit's type determines what types of resources it may have, in accordance with the game design variables *nrst* and *iars*. A unit can never receive enemy resources, but in addition IDAHEX prevents it from receiving friendly resources of a type it may not have. The game designer should tell the players which types of resources each type of unit is allowed to have. He should also tell them *toe*(i,j)--the planned quantity of type j resources in a type i battle unit--for each unit type i and resource type j.

Each cell is "owned" by either Red or Blue. The ownership of every cell at the start of the game is declared by the game design data. Suppose a given cell is owned by Red; ownership changes to Blue when a Blue battle unit assumes a hold posture in the cell without opposition, or when Red units holding the cell are defeated in an engagement and the victorious Blue attackers are allowed to occupy it. Ownership changes from Blue to Red according to an analogous rule.

3. MANEUVER

A task force is a collection of one or more battle units--"task force elements" --that have the same status (location, posture, and objective) and will continue to have the same status as long as they remain in the task force. The elements of a task force must all belong to the same side. A task force is identified by a positive integer, which bears no relation to its elements' identification numbers.

3.1 EVENT SEQUENCING

A task force's change of status is always caused and directed by an "order". Sometimes, orders are generated by IDAHEX; usually, they are input by the players. An order has two components: the desired objective and the desired posture. Associated with an order there may be a "start time". the earliest time at which execution of it should begin. A task force can not always achieve the desired posture and desired objective directly. For example, it can not attack units in an adjacent cell without first moving there. Execution of an order is a process that may span time and may take the task force through a sequence of statuses. The time required to go from one status to another may be 0, but the task force still enters every status in the sequence. For fixed values of the game design variables *pmapup* and *pmapdn*, a task force's current status and the order it is executing uniquely determine its next status. Figure 3.1 of Volume 2 shows precisely how. This manual indicates only the essentials.

Referring to Figure 3.1 of this volume, suppose a task force is located in cell 6 in posture 1, and suppose its order declares cell 9 to be the desired objective and a certain hold posture to be the desired posture. In essence, the task force is under orders to change location to cell 9 and assume a hold posture there. The task force will enter statuses in the following sequence:

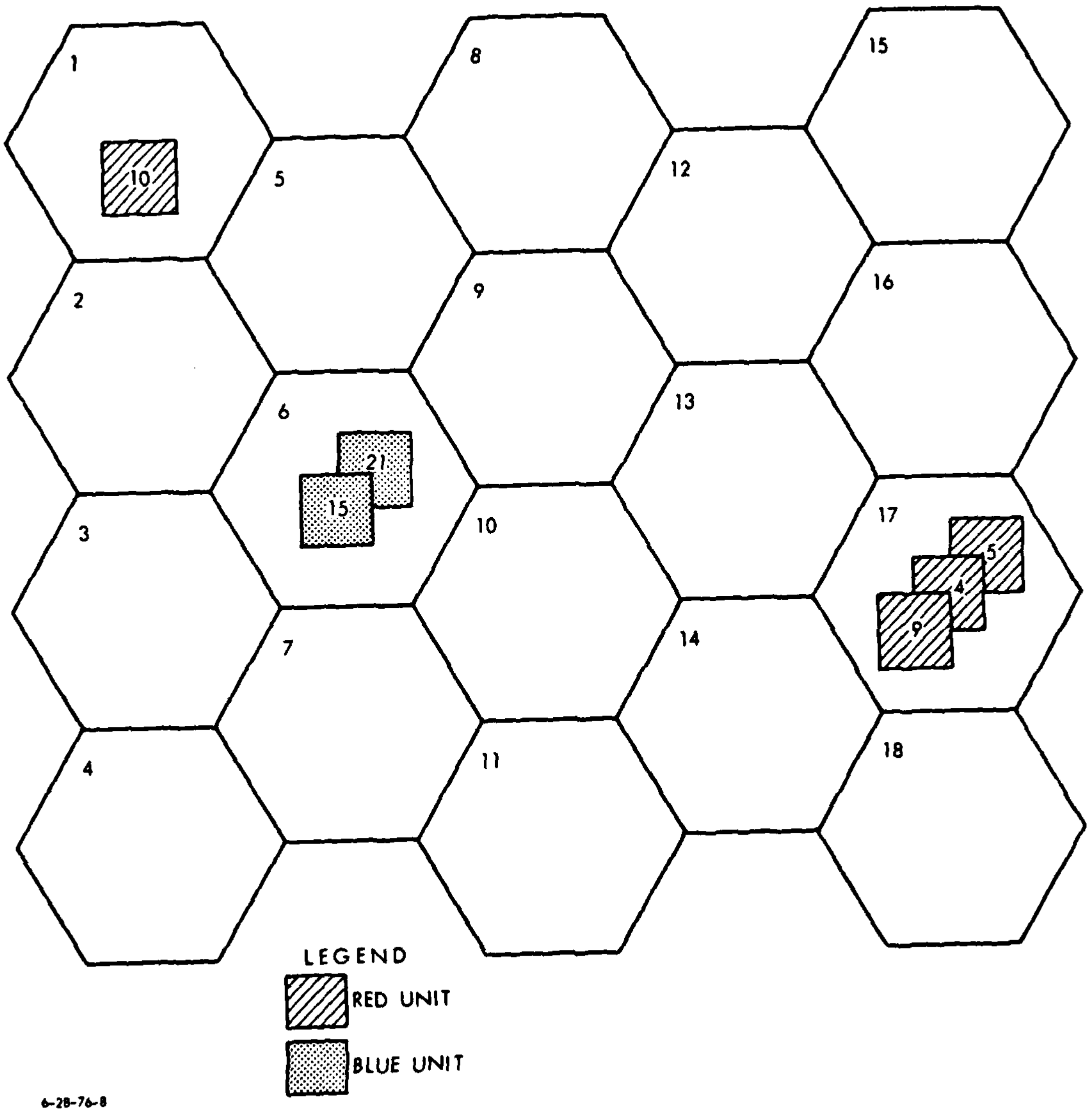

Figure 3.1. AREA OF WAR WITH BATTLE UNITS

location	posture class	objective
6	2	9
6	3	9
6	4	9
9	1	9

That is, it will first disengage from any combat in its present location, then move to the objective, then attack any enemy units in the objective, and finally, enter the objective. It will enter posture class 2 (disengagement) even if it is not engaged, in which case the disengagement is simply a formality of model logic. It will enter posture class 4 even if cell 9 contains no enemy units at the time, in which case the attack is simply a formality of model logic, involving no combat. In order to change location, the task force must enter the posture classes disengagement, movement, attacking, and holding in that order. Completion of the sequence is not assured: the time needed to get from one status to the next status in the sequence may exceed the length of the war. In particular, the task force may take forever to get from posture class 3 to posture class 4--to complete its movement--if it lacks essential supplies or transport. It may never succeed in getting from posture class 4 to posture class 1 in cell 9 if, as can happen, it must defeat enemy units defending the cell before it can occupy the cell.

The vector variables *pmapup* and *pmapdn* are set by the game designer subject to the following restrictions:

$$
\begin{array}{rl}
20 \le \mathit{pmapup}(\mathrm{pp}) \le 29 & \text{if} \quad 10 \le \mathrm{pp} \le 19 \\
30 \le \mathit{pmapup}(\mathrm{pp}) \le 39 & \text{if} \quad 20 \le \mathrm{pp} \le 29 \\
40 \le \mathit{pmapup}(\mathrm{pp}) \le 49 & \text{if} \quad 30 \le \mathrm{pp} \le 39 \\
10 \le \mathit{pmapup}(\mathrm{pp}) \le 19 & \text{if} \quad 40 \le \mathrm{pp} \le 49
\end{array}
$$

$$
\begin{array}{rl}
-10 \le \mathit{pmapdn}(\mathrm{pp}) \le -1 & \text{if} \quad 10 \le \mathrm{pp} \le 19 \\
40 \le \mathit{pmapdn}(\mathrm{pp}) \le 49 & \text{if} \quad 20 \le \mathrm{pp} \le 49
\end{array}
$$

The positive posture classes form a cyclic group, and *pmapup*(pp) is the posture a task force enters when it transitions to the next higher posture class: from posture class 1 it goes to 2, from 2 to 3, from 3 to 4, and from 4 to 1. The variable *pmapdn* is not used to take a task force from its present posture to the next lower posture class--that is generally illegal. Rather, it tells what posture a disengaging, moving, or attacking task force enters when it aborts the disengagement, movement, or attack and attempts to revert to a hold posture in its present location.

The order that a task force is executing implies whether or not the task force should transition to another posture class. If not (if the order implies it should assume another posture in the same posture class), then it will enter this new posture directly (but possible after some delay).

To get specific examples of status sequencing, suppose there are four hold postures, one disengagement posture, two movement postures, and three attack postures, and *pmapup* and *pmapdn* are defined as follows:

pp	*pmapup*(pp)	*pmapdn*(pp)
10	20	-10
11	20	-10
12	21	-10
13	21	-10
14	21	-10
20	30	42
21	31	42
30	40	42
31	41	42
40	11	42
41	12	42
42	14	42

The preceding assignments are motivated by the following interpretations of the postures:

10	standard defense
11	halted, dispersed off-road
12	halted, mainly on roads
13	prepared for transferring resources to other units (*itrfp* = 13)
14	hasty, disorganized defense
20	standard disengagement
21	disengagement mainly by road
30	tactical march
31	administrative march
40	standard attack
41	attack from administrative march
42	hasty, disorganized attack

Presumably, the ground combat attrition data make a unit less effective on defense in posture 12 than posture 11, and less effective in posture 11 than posture 10. Likewise, an attacker should be less effective in posture 41 than posture 40. Based on the above values of *pmapup* and *pmapdn* and the area of war in Figure 3.1, Table 3.1 shows the sequences of statuses induced by various orders. The last example in the table depicts a task force aborting an attack and reverting to a hold posture at its present location.

Table 3.1. EXAMPLES OF STATUS SEQUENCES

present location	present posture	present objective	desired posture	desired objective	Sequence of Statuses		
					location	posture	objective
6	10	6	10	6	6	20	9
					6	30	9
					6	40	9
					9	11	9
					9	10	9
6	11	6	31	9	6	20	9
					6	30	9
					6	31	9
6	31	9	10	9	6	41	9
					9	12	9
					9	10	9
6	31	9	40	9	6	41	9
					6	40	9
6	0	6	10	6	6	10	6
6	40	9	10	6	6	42	6
					6	14	6
					6	10	6

The preceding configuration of postures can be simplified at the risk of oversimplifying: let there be just two hold postures, one disengagement posture, one movement posture, and one attack posture, and define *pmapup* and *pmapdn* as follows:

$$\textit{pmapup}(\mathrm{pp}) = \begin{cases} 20; & 10 \le \mathrm{pp} \le 19 \\ 30; & 20 \le \mathrm{pp} \le 29 \\ 40; & 30 \le \mathrm{pp} \le 39 \\ 10; & 40 \le \mathrm{pp} \le 49 \end{cases}$$

$$\textit{pmapdn}(\mathrm{pp}) = \begin{cases} -10; & 10 \le \mathrm{pp} \le 19 \\ 40; & 20 \le \mathrm{pp} \le 49 \end{cases}$$

Suppose the game designer has designated posture 11 as the transfer posture (by setting *itrfp* = 11). Refer again to Figure 3.1. A task force in a hold posture in cell 6 whose desired posture is 11 and desired objective is 9 would go through the following sequence of statuses:

location	posture	objective
6	20	9
6	30	9
6	40	9
9	10	9
9	11	9

When the unit achieves posture 11, it will be ready and able to transfer resources to other units located in cell 9. A task force whose

location = 6,
posture = 40,
objective = 9,

and whose

desired posture = 10,
desired objective = 6,

would go through the following sequence:

location	posture	objective
6	40	6
6	10	6

Thus, the task force aborts an attack and goes directly into the standard hold posture, in contrast to the last example in Table 3.1; there is no "disorganized defense" posture in which to put it.

Because that can happen whenever the game designer selects a skeleton configuration of postures, IDAHEX provides another way of reducing a task force's defensive capability in that situation. A unit's defensive capability in a hold posture--specifically, its resources' vulnerability to fire from enemy battle units--depends on how long it has had to prepare its defense. This "defense preparation time" is computed as the current time minus the virtual time at which it entered posture class 1 in its location. On output, the virtual time at which a unit entered its present posture class is labeled "t entry", "tentry", or "time of entry". The time is virtual, rather than actual, because IDAHEX may penalize a unit's hasty reversion to a hold posture by setting the virtual time of entry later than the time it actually entered posture class 1, with the result that its defense preparation time may be negative.

In every example of task force movement thus far, the objective has been a cell adjacent to the task force's location, but IDAHEX's event sequencing logic does not require that. A task force may receive an order stating a desired objective that is not adjacent to its present location. The task force will be able to execute the order only if it moves entirely by air.

3.2 EVENT SCHEDULING

As Section 3.1 explains, a task force's execution of an order involves going through a sequence of one or more statuses. Associated with any change of status is a delay time. The task force stays in its present status (present location, present posture, present objective) a length of time equal to the delay and then enters the next status (next location, next posture, next objective) in the sequence. This subsection only mentions the factors that affect the delay; Section 3.2 of Volume 2 explains fully.

3.2.1 Changing Posture within a Positive Posture Class

Suppose that the task force's present posture class is positive, and that its next posture class is the same as its present posture class and its next objective is the same as its present objective--i.e., it is going to a different posture in the same class. The delay is given by the game design variable *ptran*: *ptran*(i,j,k) is the time required to go from the j-th posture to the k-th posture within posture class i.

3.2.2 From a Hold Posture to a Disengagement Posture

In this case the present posture is a hold posture and the next posture is a disengagement posture. The delay is 0.

3.2.3 Disengagement

In this case, the present posture is a disengagement posture and the next is a movement posture, and the next objective is the same as the present objective. The delay is most simply interpreted as the time required to break contact with the enemy, but in reality a force being pursued by the enemy might never break contact completely. The delay is better interpreted as the amount by which contact with the enemy increases the time needed for the task force to relocate from its present location to its objective. If the task force is not engaged, the delay is, of course, 0.

If the task force is engaged, two situations must be distinguished: (1) there are friendly units in hold postures in the task force's present location; (2) there are none. In the first situation, the friendly units are assumed to prevent the enemy units with which the task force is engaged from pursuing it, and therefore the delay does not depend on the time required for the task force to move to its objective; the delay represents only the time required to break contact with the enemy in the present location. Of course, some types of units are more adept at breaking contact than others; because the elements of the task force operate together, the task force can only disengage as fast as its slowest element. In the second situation, the absence of friendly units that could act as a rearguard for the task force permits the enemy units engaged with it to pursue it. Instead of describing the task force's movement and its continuing engagement as contemporaneous processes, IDAHEX extends the task force's disengagement delay (before it moves to its objective). That is, its delay equals the delay it would have experienced in the first situation plus an additional term that is proportional to the task force's anticipated movement delay. The additional term may be interpreted as the time during which the task force interrupts its movement to the objective to deal with enemy pursuit.

3.2.4 Movement

In this case, the present posture is a movement posture, the next posture is an attack posture, and the next objective is the same as the present objective. Recall from Section 3.1 that before a task force changes location it enters an attack posture with the new location as the objective, even if the objective contains no enemy units (in which case the "attack" is purely a logical formality). The "movement delay" is the length of time the task force remains in the movement posture before entering the attack posture. It represents the time required to move (unopposed) from the present location to the objective. If the objective contains enemy units in hold postures, the task force must attack and defeat them in order for its location to become that cell, but before it can attack there it must move there.

The task force is not considered to have arrived at its location until all its resources have (equivalently, its faster resources are assumed to travel slowly enough not to outdistance other resources), but its resources may travel by different modes--cross-country/road, rail, sea, or air. Resources of a given type can travel under their own power in only one mode. In any movement posture, some types of resources are moving independently while other types of resources may be carried. Carried resources, or "passengers", use independently moving resources for transportation; for example, small arms and personnel might ride in trucks or aircraft, tanks might ride in trains or ships. Sometimes passengers do not rely completely on independently moving resources for transportation; for example, personnel may walk when they are not riding in trucks. But passengers use transportation to the extent it is available. Passengers may ride on any independently moving resources that can accomodate them. But only those types of resources designated "ferries" may move backward and forward between the location and the objective--carrying passengers when they move forward, then dropping them off and going back for more. The task force's movement posture and the types of units it contains determine which types of its resources are independently moving; which of these are ferries; and which types are passengers.

The movement delay is calculated as the sum of three distinct delays, d0, d1, and d2. The delay d0 is either 0 or $+\infty$ (in which case the movement delay is infinite). It enforces prohibitions against certain movements by making the time needed to accomplish them infinite. The delay d1 is proportional to the distance traveled, while d2 represents time spent crossing barriers. The movement delay is calculated according to the procedure outlined below.

Step 0. Initially, let $d0 = d1 = d2 = 0$. If the task force's objective (the cell to which it is moving) is nonexistent or inactive, let $d0 = +\infty$ and go to Step 7.

Step 1. Classify each type of the task force's resources as independently moving resources or passengers. Determine the types of independently moving resources that are ferries.

Step 2. If the task force's objective is not adjacent to its location, is the location of an active enemy unit, and is not owned by the task force's side, then let d0 = $+\infty$ and go to Step 7. (This precludes the situation in which an attacking airborne task force is engaged with enemy units in a cell not adjacent to its location.)

Step 3. If the task force lacks supplies it needs in order to move, let d0 = $+\infty$ and go to Step 7.

Step 4. If some type of passenger resource is physically incapable of being transported by the independently moving resources (e.g., ships cannot be loaded on planes), let d1 = $+\infty$ and go to Step 7. Find cap, a number that measures the independently moving resources' aggregate capacity to carry the passengers, and burd, a number that measures the burden the passengers impose on this capacity (burd = 0 if there are no passengers). If cap = 0 and burd > 0, let d1 = $+\infty$ and go to Step 7.

Step 5. Find the speed at which each type of the task force's resources can move, under their own power, from the location to the objective, considering the movement posture as well as the trafficability of the route. (See below.) If this is 0 for some type of independently moving resources, let d1 = $+\infty$ and go to Step 7.[1]

If burd $\leq$ cap, let d1 = D/S, where D is the straightline distance from the center of the cell that is the task force's location to the center of the cell that is its objective and S is the minimum speed of the independently moving resources, and go to Step 6. If burd > cap > 0, the ferries must make multiple trips to get all the passengers and themselves to the objective, complicating the calculation of d1.[2]

Step 6. Now d2, the delay attributable to a barrier, is calculated. (At this point, d2 = 0.) If none of the independently moving resources are traveling on land, go to Step 7: no barrier can affect them. If there is no movement barrier between the task force's location and its objective, go to Step 7.

Certain types of movement barriers interrupt rail traffic--a river without a railroad bridge, for example. These barriers

[1] If the objective is not adjacent to the location, the speed of resources that can not fly is defined to be 0. Therefore, the task force can go directly to a nonadjacent cell only if all its independently moving resources can fly.

[2] See Volume 2, Section 4, for a complete explanation.

imply the rail link has been severed. If such a barrier exists between the location and the objective, and if some of the independently moving resources are traveling by rail, then let d2 = +∞ and go to Step 7. (Recall that each type of resources has a characteristic model of travel--cross-country/road, rail, air, or sea--which they use whenever they move independently.) If no such barrier exists and all the independently moving resources that are traveling on land are traveling by rail, go to Step 7: there is no barrier delay.

In the remaining cases, the task force's barrier delay is assessed as the maximum delay of any element with resources traveling by road or cross-country. Any such element's delay depends upon its unit type, the task force's movement posture, and the movement barrier type.

Step 7. Set the movement delay = d0 + d1 + d2. End.

Recall that d1 depends on the speeds of the various types of resources--the speeds at which they could move, without assistance from other resources, from the task force's location to its objective in its movement posture. Consider resources of a given type. Their type determines their mode of travel. If they are flying, their speed is given directly by the game design data (*vair*). If they are not flying and the objective is not adjacent to the location, then their speed is defined to be 0. Suppose they are not flying and the objective is adjacent to the location. If they are sailing, their speed is given directly by the game design data (*vsea*), except that their speed is defined to be 0 if the objective is not a water cell. If they are moving by rail, their speed depends upon their type and the type of rail link between the location and the objective; if there is no rail link, or one exists but it is damaged, then the resources' speed is defined to be 0.[1] In the remaining case, the resources are using some combination of road and cross-country movement (possibly all road or all cross-country). Their road movement rate depends on their type, the type of road link between the location and the objective, and the task force's movement posture; their road movement rate is defined to be 0 if there is no road link. Their cross-country movement rate depends on their type, the environments in the location cell and the objective cell, and the movement posture. The fraction of the distance that they travel cross-country, instead of on roads, depends upon their type, the type of road link between the location and the objective, and the movement posture. If there is no road link, the distance traveled by road is necessarily 0; if there is a road link but it is damaged, the resources move cross-country more than they otherwise would. The resources' (average) speed

[1] Section 8 explains how railroads and roads can become damaged.

depends upon their road movement rate, their cross-country movement rate, and the fraction of the distance they travel by road.

The environment, road, rail, and movement barrier types assumed in the calculation of a movement delay may change at the start of a cycle. Consequently, IDAHEX may re-calculate a task force's movement delay at that time. Also, a task force's movement may be aborted because it runs out of required supplies.

3.2.5 Attack

Suppose the present posture is an attack posture, the next posture is a hold posture, and the objective is not the same as the present location. If the task force's objective contains no enemy units in hold postures, the delay is 0. Otherwise, the delay is indefinite: it depends on the course of combat.

3.2.6 Reorientation to the Present Location

Two cases must be distinguished. In the first case, a disengaging, moving, or attacking task force whose present objective is not its present location seeks to revert to a hold posture in its present posture (perhaps only momentarily, before moving in a different direction). Then its next posture class is 4, and its next objective is its present location. The delay is 0. And then the second case arises.

In the second case, the task force is in an attack posture, but its objective is its present location. Its next posture class is 1, and its next objective and next location are its present location. The delay is 0.

3.2.7 Activation

Suppose a task force consisting of inactive units is given an order to active them, an order stating a positive desired posture class. The delay is infinite if the present posture class is -1: a destroyed unit can not come back to life. The delay is 0 if the present posture class is 0.

3.2.8 Transition to or within a Nonpositive Posture Class

In this case the next posture class is -1 or 0. The delay is 0.

A player can order a task force in posture class 0 to a different location, which it can attain instantaneously if the new location is not owned or occupied by the enemy, and then activate its elements by ordering it into posture class 1. This capability allows reinforcements to arrive in the area of war at an appropriate place and time. To avoid unrealistic events the player must follow instructions from the game designer stating where and when inactive units may be activated.

A player can also order a task force in posture class -1 to posture class 0 (and then to posture class 1), but since there is normally no reason for a unit to enter posture class 0 from another posture class, IDAHEX warns the game designer if that happens.

3.3 TACTICAL SITUATIONS

Maneuver of opposing battle units into proximity may precipitate tactical situations in which it is advisable or essential that IDAHEX issue new orders to existing task forces or create new task forces and issue them orders. In some cases the sole purpose of the orders is to respond immediately to a local situation that one of the players probably failed to foresee. Often, the orders have the additional purpose of resolving opposing forces' conflicting aims.

The player should watch for one tactical situation with potentially severe consequences. If an attacking task force is actually engaged and its location is occupied by the enemy, then it is destroyed--IDAHEX orders it into posture class -1. The penalty is severe, but is the only acceptable way of resolving the situation: the task force controls neither its location nor its objective and therefore has no place to exist. IDAHEX takes extensive precautions to avert this situation. If a task force is threatened with destruction because an enemy task force is about to occupy its location, IDAHEX orders the task force to revert to holding its location. If the task force succeeds in doing that, it still incurs a penalty: its defensive capability may be degraded because the hold posture it enters may represent a hasty, disorganized defense and because its defense preparation time (see below) may be zero or even negative.

The defense preparation time of a battle unit in a hold posture is a measure of how prepared the unit is to defend its location. It is defined as the current time minus the unit's virtual time of entry into posture class 1. Ordinarily, when a unit enters posture class 1, its virtual time of entry is set equal to the actual time. But when a unit in a disengagement, movement, or attack posture reverts to a hold posture in its location while an enemy unit directly threatens to seize its location from the flank or rear (not from the cell that was the objective of its disengagement, movement, or attack), then the unit's virtual time of entry into posture class 1 is set ahead of the actual time by an amount that measures how far out of position the unit was before it resumed holding its location.

4. THE COMMANDS

Periodically, the Red player and the Blue player may input commands to IDAHEX. A command is an instruction to battle units or a request for information. IDAHEX prevents a player from issuing instructions to enemy units or obtaining the enemy player's instructions to his units. IDAHEX signals a player when it is ready to receive commands from him. The Red player may go before the Blue player, but that is purely a formality: both players' battle units begin executing their instructions simultaneously. In no way do the forces take turns.

IDAHEX signifies that is ready to receive a command by writing "Enter command." on the player's terminal. The player replies by entering a character string enclosed in quotation marks. IDAHEX examines the reply to identify the command. If complete specification of the command requires more inputs from the player, IDAHEX writes prompting sentences on the terminal. ("Enter command." is the initial prompting sentence.) The player must reply to a prompting before IDAHEX will proceed. Each reply is a character string enclosed in quotation marks.[1] It may occupy more than one line of input. A reply may contain no items: it consists only of blanks, or it is empty. Examples:

```
"    "
""
```

Or a reply may contain exactly one item. Examples:

```
"oneword"
" oneword"
"29.2"
" 29.2 "
"17"
```

Or it may contain several items. Examples:

[1] It may be permissible to omit the quotation marks if the reply contains exactly one item.

```
"firstword secondword"
"firstword,secondword"
"29.2 17, 12.1 , 15"
"word,17,12.1"
```

In general, two consecutive items can be separated by one or more blanks, by a comma, or by a comma together with one or more blanks. Hence, each of the following replies contains exactly two items:

```
"7 45"
"7,45"
"7, 45"
"7 , 45"
"  ,  "
","
```

In the last two replies the two items are both null--i.e., the items contain no nonblank characters. IDAHEX interprets a null item as a character string of blanks if it expects the item to be a character string, and as the number 0 if it expects a number. As one might infer, IDAHEX maps a reply into a "target list" of items it expects to receive. A target list is actually a list of variables to which the reply implicitly assigns values.

Each non-null item in a reply--each item containing a nonblank character--is classified as either a number or a word. A non-null item is a number if and only if its first nonblank character is a decimal digit, a decimal point, or a minus sign (-); a non-null item is a word if and only if it is not a number. *All the words in a reply must precede all the numbers.*

Each variable in a target list is classified as either a character-string variable or an arithmetic variable. It is a character-string variable if and only if it is intended to receive a word; it is an arithmetic variable if and only if it is intended to receive a number.

Each item in a reply is assigned to the corresponding variable in the target list, its "target variable". When a null item is assigned to a variable, the variable's value becomes a character string of four blanks if the variable is a character string variable, and its value becomes 0 if it is an arithmetic variable. If, in comparing the reply with the target list, IDAHEX discovers that reply items are missing, it implicitly inserts a null item to take the place of each missing item. Hence, the null reply ("") gives every character string variable in the target list the value "   " and every arithmetic variable in the target list the value 0.

A^tually, only the first four characters of a word are assigned to its target variable. Therefore, a player may truncate any word to its first four characters. In particular, the command names "mission", "deliver", "transfer", "cancel", "abandon", and "display" may be abbreviated as "miss", "deli", "tran", "canc", "aban", and "disp".

The sequel generally shows the format of a reply in a line labeled "Reply" or "Initial reply". *Any item not enclosed between a "less-than" character (<) and a "greater-than" character (>) must appear exactly as shown, except that a word may be truncated to four characters. An item enclosed between a "less-than" character and a "greater-than" character is an argument, to be replaced by an appropriate word or number without the "less-than" and "greater-than" characters. It may be omitted, but then every item following it in the reply must also be omitted.* Omitting it has precisely the same effect as including it in the reply as a null item (which, if it should be a number, has the same effect as including it in the reply as the number 0).

The "initial reply in the command sequence" is the player's reply to the prompting sentence "Enter command." A line labeled "Initial reply" shows its format.

IDAHEX's command repertoire is occasionally expanded. The program that the players are using may support commands in addition to those described in this section. The command repertoire can be learned by invoking the help command.

4.1 ASSIGNING AND REVISING MISSIONS

As Section 3.1 explains, a task force's change of status is always caused and directed by an order. A mission is a sequence of orders. Every task force has a mission, and every mission is assigned to exactly one task force (but two task forces may have identical missions). A task force and its mission are both identified by the same positive integer, which is assigned by IDAHEX or the player when the task force is created. The orders comprising a mission are stored in a pop-up stack and are executed in sequence, from the top to the bottom. The order at the top of the stack is termed the active order. When execution of it is completed, it is removed from the stack, and the next order, if any, pops to the top. If a start time is associated with the active order, execution of the order does not begin until the game time equals or exceeds the start time.

A mission is created or modified by the mission command:

Initial reply: "mission <number1>"

Suppose number1 happens to be the identification number of an existing task force (which must be positive). The player is then modifying the task force's mission. Its present mission is erased, and the player constructs a new one by entering a sequence of orders.

Prompting: Enter orders.

Reply: "<dobj> <dpost> <strt>"

Actually, the player enters a sequence of replies. The first reply defines the first order in the mission, the second reply defines the second order, and so on. Recall that an order has two components: a desired objective and a desired posture. The item dobj defines the desired objective (a cell number), and dpost defines the desired posture. There is an IDAHEX variable named nstart.[1] The player may specify a start time for each of the first nstart orders in a mission; the argument strt is ignored if it appears in a reply after the first nstart replies. Of course, if strt is omitted from one of the first nstart replies, the corresponding order's start time is taken to be 0.[2]

More than one reply to the prompting sentence "Enter orders." may occur on the same line, but if so, at least one blank should separate them.

The player signals IDAHEX that he has input the final order in the mission by entering a reply that defines the desired objective as a nonpositive number (as the null reply ("") does). The dummy order defined by this reply does not become part of the mission.

If the player is modifying an existing mission, those are his only inputs. Alternatively, suppose he is creating a new task force and corresponding mission--which is true if and only if number1 is not the identification number of an existing task force (or it is omitted from the initial reply). The identification number assigned to the task force and mission is number1 if

[1] The value of nstart is set in the IDAHEX main program, cgcm. It equals 1 in Version 2.

[2] In that case execution of the order will begin as soon as execution of the preceding order, if any, is completed--assuming, as should be the case, that the game time is nonnegative.

number1 > 0 and is not too large; IDAHEX chooses a number otherwise.[1] IDAHEX requests the player to "Enter orders.", which he does just as before. After he has entered the final order, he must list the elements of the task force, in any sequence:

Prompting: List task force.

Reply: "<unit1>, <unit2>,..., <unitn>"

Each reply item identifies a task force element by its battle unit number. Any unit listed that already belongs to a task force is automatically detached from it.

The examples below are based upon the area of war in Figure 3.1 and the configuration of postures assumed by Table 3.1, namely:

pp	*pmapup*(pp)	*pmapdn*(pp)
10	20	-10
11	20	-10
12	21	-10
13	21	-10
14	21	-10
20	30	42
21	31	42
30	40	42
31	41	42
40	11	42
41	12	42
42	14	42

The examples assume nstart = 1 and therefore specify a start time only for the first order in a mission.

Example 1. Suppose units 4 and 9, located in cell 17, are in posture 12. In the following communications with IDAHEX, the Red player constitutes them as a task force and assigns a mission.

[1] A mission's identification number may not exceed nmnmax, whose value is fixed in the entry point cccm.

```
Enter command.
"mission"
Enter orders.
"16 12"
"12 30"
"12 10"
""
List task force
"9 4"
```

Each line after the prompting sentence "Enter orders." states an order: the first number is the desired objective (i.e., the identification number of the cell that is the desired objective), and the second number is the desired posture. Because nstart ≥ 1, the reply "16 12" is mapped into a list of three items, the third being the start time. Since the reply contains only two items, the start time is defined to be 0. The mission implies the following sequence of statuses for the task force:

location	posture	objective	
17	21 (disengaging)	16	
17	31 (admin. march)	16	
17	41 (attack from 31)	16	
16	12 (halted)	16	Order 1 finished.
16	21 (disengaging)	12	
16	31 (admin. march)	12	
16	30 (tactical march)	12	Order 2 finished.
16	40 (standard attack)	12	
12	11 (halted, dispersed)	12	
12	10 (standard hold)	12	Mission finished.

Example 2. Suppose task force 38, consisting of unit 27, is moving tactically from cell 6 to cell 5 in accordance with a mission whose active order is "desired objective = 5, desired posture = 10", and whose next order is "desired objective = 1, desired posture = 10". The player directs that the movement to cell 5 continue, but thereafter the task force should go to cell 2 instead of cell 1:

```
Enter command.
"miss 38"
Enter orders.
"5 11"
"2 10"
""
```

The modified mission implies the following sequence of statuses for task force 38:

location	posture	objective	
6	40 (standard attack)	5	
5	11 (halted)	5	Order 1 finished.
5	20 (disengaging)	2	
5	30 (tactical march)	2	
5	40 (standard attack)	2	
2	11 (halted)	2	
2	10 (standard hold)	2	Mission finished.

The original mission implied next location = 6, next posture = 40, next objective = 5. *Since the modified mission implies the same next status as the original one, the time at which the task force will achieve its next status is not rescheduled.* The player could have achieved the same sequence of statuses for unit 27 by first canceling mission 38, using the cancel command explained below, and then entering the following command sequence:

```
Enter command.
"miss 38"
Enter orders.
"5 11"
"2 10"
""
List task force.
"27"
```

There would be only one difference in the results: because the cancel command destroys all record of the original mission, and a new task force has been created (albeit with the same identification number and same composition), the movement delay is re-evaluated, and the time at which unit 27 completes its movement may be rescheduled. In general, *it is wise to modify the mission of a disengaging or moving task force, through the mission command, rather than canceling it and recreating the task force with a new mission.*

Example 3. Suppose task force 39 is moving from cell 6 to cell 5. In the following communications, the player directs it to move instead to cell 9:

```
Enter command.
"miss 39"
Enter orders.
"9,10"
""
```

The new mission implies the following sequence of statuses for task force 39:

location	posture	objective	
6	42 (hasty attack)	6	
6	14 (hasty defense)	6	
6	21 (disengaging)	9	
6	31 (admin. march)	9	
6	41 (attack)	9	
9	12 (halted)	9	
9	10 (standard hold)	9	Mission finished.

Example 4. Suppose unit 21 is in posture class 0 in cell 6. The player assigns a mission to the task force consisting of unit 21:

```
Enter command.
"miss"
Enter orders.
"6 13 1.51"
"9 13"
""
List task force
"21"
```

The mission implies the following sequence of statuses for unit 21:

location	posture	objective	
6	10 (standard hold)	6	
6	13 (transfer)	6	Order 1 finished.
6	21 (disengaging)	9	
6	31 (admin. march)	9	
6	41 (attack)	9	
9	12 (halted)	9	
9	13 (transfer)	9	Mission finished.

The first change of status will not occur until time 1.51. The example illustrates one way of accomplishing resupply and replacement: if new resources should enter the area of war in cell i at time x, the game design data should incorporate them into a unit whose initial location is i and initial posture class is 9; the player whose side should receive the resources can issue a mission command to activate the unit at time x.

Example 5. Assume unit 21 is in posture class 0. In the following communications with IDAHEX, the Blue player activates unit 21 in cell 8 instead of its present location, cell 6.

```
Enter command.
"miss"
Enter orders.
"8 0"
"8 10"
""
List task force.
"21"
```

The mission implies the following sequence of statuses for unit 21:

location	posture	objective	
8	0 (inactive)	8	Order 1 finished.
8	10 (standard hold)	8	Mission finished.

The first change of status is scheduled to occur immediately, assuming that the current time is nonnegative.

Thus, a player can activate one of his units in a cell different from its initial location; to do so, he must first change its location while it remains in posture class 0. This capability is necessary since the location where a package of supplies and replacements should become available might depend on the course of the game. Indeed, IDAHEX prohibits activation of a unit in a cell owned by the enemy or containing enemy units. And it might be convenient to design the game so that supplies and replacements originate in corps, army, or front depots, which relocate to keep up with the combat forces, rather than fixed theater depots. A player could use the capability to change inactive units' locations in order to cheat, relocating units wherever he pleased. Therefore, IDAHEX places an advisory message in a file intended for the game designer whenever an inactive unit changes location.

4.2 DETACHING A UNIT FROM A TASK FORCE

The detach command detaches a battle unit from a task force. The unit then has no mission. There is no effect on the mission of the remaining task force or the time at which the task force is scheduled to enter its next status.

Initial reply: "detach <unitnbr>"

The argument unitnbr is the number of the battle unit to be detached.

Example. The player commands IDAHEX to detach unit 4 from the task force to which it belongs, if any:

```
Enter command.
"deta 4"
```

It is not necessary to give a detach command to detach a unit from a task force in preparation for including it in a new one: naming a unit as an element of a new task force in a mission command automatically causes it to be detached from any task force to which it already belongs.

4.3 ATTACHING A UNIT TO A TASK FORCE

The attach command attaches a battle unit to an existing task force. If the unit belongs to another task force, it is detached before being attached to the designated task force. It must have the same status as the designated task force and must, of course, belong to the same side.

Initial reply: "attach <unitnbr> <tfnbr>"

The argument unitnbr is the number of the battle unit to be attached, and tfnbr is the number of the task force to which it should be attached.

Attaching a unit to a task force may push back the time at which the enlarged task force is scheduled to enter its next status. Let t1 be the time at which the original task force is scheduled to enter its next status. Let t2 be the time for which the enlarged task force's transition from its present status to its next status would be scheduled if it were just beginning the transition at the present time. The enlarged task force is scheduled to enter its next status at time t1 or t2, whichever is greater.

Example. The player commands IDAHEX to attach unit 4 to task force 17:

```
Enter command.
"atta 4,17"
```

4.4 TRANSFERRING RESOURCES

Three different commands accomplish transfers of resources from one set of units to another set of units. To use the transfer and delivery commands, a hold posture must be reserved as the transfer posture. (Its number is given by the game design datum *itrfp*.) The transfer command and the delivery command can be used to transfer resources from one set of units, called the givers, to another set, called the takers, provided the givers and takers all belong to the same side, they all have the same location, and the givers are all in the transfer posture. A taker may be in any of the postures from 10 through 49, including the transfer posture. A unit may be both a giver and a taker. A taker can accept resources in excess of the planned quantity for a unit of its type, even if that quantity is 0, but it cannot accept resources it is prohibited from having. (See Section 2.2.2.)

The send command can be used to transfer resources from one set of units (the givers) to another set (the takers) provided the units all belong to the same side. The units may have different locations and the givers may be in any posture from 10 through 49. Use of the send command normally occurs only in games where the logistics system is not explicitly played.

4.4.1 The Delivery Command

The delivery command permits the player to arrange for a transfer of resources that will occur automatically, at the earliest possible moment. The command creates a "delivery order" (not to be confused with the orders in a mission). A delivery order has four components: (1) the task force designated to deliver the resources; (2) the delivery destination; (3) the relative size of the delivery; (4) the intended recipients of the delivery. The delivery destination is the cell where the delivery is to occur. The relative size is a number between 0.0 and 1.0, inclusive, that indicates how much should be transferred. Once created, a delivery order continues to exist until it is executed or it is canceled by the player.

The "delivery task force", the one designated to make the delivery, must exist when the delivery order is created and when it is executed. Two or more delivery orders may name the same delivery task force, but their delivery destinations should differ. If two delivery orders designate the same delivery task force and the same delivery destination, IDAHEX arbitrarily selects one of them. The list of intended recipients may be empty.

Initial reply: "delivery <tfnbr> <cellnbr>"

The argument tfnbr is the delivery task force's number, and cellnbr is the delivery destination's cell number. Next, the player must indicate the size of the delivery.

Prompting: Enter relative size of delivery.

Reply: "<dlsize>"

The item dlsize should be a fixed decimal constant between 0 and 1. Finally, the player lists the delivery's intended recipients. (The list's purpose is explained below.)

Prompting: List recipients.

Reply: "<id1>, <id2>,..., <idn>"

A reply containing no items defines the empty list. Each item--id1,...,idn--must identify either a friendly battle unit or a friendly task force. An item identifying a unit is simply the unit's number; an item identifying a task force is the sum of the task force's number and 10000.

Suppose task force m has just entered the transfer posture in cell i. IDAHEX must decide whether the transfer of resources will be governed by a transfer command (see the next subsection) that the player will issue later or by a delivery order. *IDAHEX infers that the player intends to issue a transfer command, and therefore makes no delivery of resources at this time, if either of the following conditions holds:*

(1) with this change of status, the task force has accomplished its mission;

(2) with this change of status, the task force has completed execution of the active order in its mission, and its new active order has a start time that exceeds the current time.

If neither condition holds, IDAHEX searches for a delivery order--one whose delivery task force is m *and delivery destination is* i. *If none is found, a delivery order is generated: it designates task force* m *as the delivery task force and cell* i *as the delivery destination, fixes the delivery's relative size at* 1.0, *and leaves the list of intended recipients empty.* A generated delivery order is treated in the same way as a delivery order created by the delivery command. Consequently, there is no reason for a player to invoke the delivery command except to make the relative size less than 1.0 or to name intended recipients.

When IDAHEX executes a delivery order, it identifies the givers and the takers. The givers are the elements of the delivery task force. To find the takers, the list of intended recipients is expanded by replacing each task force mentioned in it with the task force's elements. If the resulting list is not empty, the takers consist of every active, friendly unit in the list whose location is the delivery destination; if the list is empty, the takers consist of every active, friendly unit whose location is the delivery destination and whose posture is not the transfer posture.

In a resource transfer governed by a delivery order, a giver can only give resources in excess of its planned quantities. To be precise, for any k, the quantity of type k resources that a particular giver can transfer to the takers is limited to the amount by which its actual stock of type k resources exceeds its planned stock.[1] Each taker demands resources of a given type to the extent that its stock falls short of its planned stock. The total quantity of resources of a given type transferred from the givers equals min{f*S,D}, where f is the delivery's relative size, S is the maximum amount that the givers can give, and D is the takers' total demand. The quantities taken from the various givers and the quantities given to the various takers are determined so as to balance the units' stocks as much as possible. Basically, a unit's stocks are perfectly balanced if its stock of each type of resources coincides with its planned stock. Section 5.2 of Volume 2 explains fully.

Example. The player arranges for a delivery of resources by task force 4 to unit 2, unit 6, and task force 30 in cell 10.

```
Enter command.
"deli 4 10"
Enter relative size of delivery.
".65"
List recipients.
"2 6 10030"
```

4.4.2 The Transfer Command

The transfer command causes an immediate transfer of resources from the givers to the takers.

Initial reply: "transfer <cellnbr>"

The argument cellnbr is the number of the cell that is the transfer location--the cell where the transfer is to occur. Next, the player identifies the units and task forces from which resources should be transferred.

Prompting: List resource givers.

Reply: "<id1>, <id2>,..., <idn>"

[1] Recall that the planned stock of type k resources in a type j battle unit is *toe*(j,k).

A reply containing no items defines the empty list, whose implications are described below. Each item in the reply identifies either a friendly battle unit or a friendly task force. An item identifying a battle unit is simply the unit's number; an item identifying a task force is the sum of the task force's number and 10000. Next, the player lists units and task forces to which resources should be transferred.

Prompting: List resource takers.

Reply: "<id1>, <id2>,..., <idn>"

A reply containing no items defines the empty list. Each item identifies either a friendly battle unit or a friendly task force as before. Finally, the player indicates the amounts of resources to be transferred from the givers to the takers.

Prompting: Enter resource index and amount to be transferred, line by line.

Reply: "<rsnbr> <amt>"

Actually, the player enters one or more replies. The first item of a reply, rsnbr, is a resource type, identified by number, and the second item, amt, is the total quantity of type rsnbr resources to be transferred, which may be nonintegral. The player's last reply must define the resource type to be nonpositive (as the null reply does), signaling that he is finished. He may decline to specify the amount to be transferred for any type of resources, or every type.

If the player's list of givers is empty, the givers consist by default of every active, friendly unit whose location is the transfer location and whose posture is the transfer posture. If his list of takers is empty, the takers consist by default of every active, friendly unit whose location is the transfer location and whose posture is not the transfer posture.

If the set of givers is identical to the set of takers, then regardless of any transfer amounts the player has specified, the units' resources are redistributed among them to balance their stocks as much as possible; basically, a unit's stocks are perfectly balanced if they coincide with its planned stocks.

Alternatively, assume the set of givers is not identical to the set of takers. Suppose, for a given i, that the player has specified the amount of type i resources to be transferred. If at least one of the takers is permitted to have this type of resources, then the amount actually transferred is the amount the player specified or the amount the givers have, whichever is less. On the other hand, suppose the player has not specified the amount of type i resources to be transferred. The amount transferred equals min{S,D}, where S and D are defined as in Section 4.4.1. In either case, the amount transferred is taken from the givers and distributed among the takers so that the units' stocks are as balanced as possible.

Example. The player commands a transfer of resources in cell 10 from unit 55 and task force 29 to the active, friendly units in cell 10 that are not in the transfer posture.

```
Enter command.
"tran 10"
List resource givers.
"10029 55"
List resource takers.
""
Enter resource index and amount
to be transferred, line by line
"3 27.45"
"8 0"
"2 12"
""
```

The amount of type 8 resources transferred will be 0.

4.4.3 The Send Command

The send command causes an immediate transfer of resources from the givers to the takers.

Initial reply: "send ⟨cellnbr⟩"

The argument cellnbr may be omitted.

Next, the player identifies the units and task forces from which resources should be transferred.

Prompting: List resource givers.

Reply: "⟨id1⟩, ⟨id2⟩,..., ⟨idn⟩"

Each item in the reply identifies either a friendly battle unit or a friendly task force. An item identifying a battle unit is simply the unit's number; an item identifying a task force is the sum of the task force's number and 10000.

If the argument cellnbr was given (and is nonzero) it is interpreted as the cell to which resources should be transferred:

the takers are defined to be every active, friendly unit in cell cellnbr. If the argument cellnbr was omitted (or was 0), the player is asked to indicate the takers:

Prompting: List resource takers.

The player's reply may have any of three different forms.

Reply (form 1): "<id1>, <id2>,..., <idn>"

In this form, the items id1,...,idn are integers. Each identifies a friendly battle unit or friendly task force as before. Alternatively, the reply may have the following form:

Reply (form 2): "interval <id1> <id2>"

(The word "interval" may be truncated to "inte".) The takers are defined to be every active, friendly unit numbered between id1 and id2, inclusive. Alternatively, the reply may have the following form:

Reply (form 3): "subset <id1> <id2>"

In this case, the takers are defined to be every active, friendly unit numbered between id1 and id2 with the exception of units the player specifically excludes:

Prompting: List units to be excluded.

Reply: "<unit_id1>, <unit_id2>,...,<unit_idn>"

Finally, the player indicates the amounts of resources to be transferred from the givers to the takers, exactly as in the transfer command. As in the case of the transfer command, he may decline to specify the amount transferred for any, or every, type of resources. The actual quantity transferred from a giver to a taker is determined according to the same rules used to interpret the transfer command.

Example. The player commands a transfer of resources from unit 100 to every active, friendly unit numbered between 1 and 65 with exception of units 14 and 17.

```
Enter command.
"send"
List resource givers.
"100"
List resource takers.
"subs 1 65"
List units to be excluded.
"14 17"
```

```
Enter resource index and amount to be
transferred, line by line.
""
```

4.5 INITIATING SPECIAL ACTIVITIES

A battle unit may be able to perform any of the following special activities:

type	special activity	abbreviation
1	barrier intensification	bi
2	railroad destruction	rrd
3	road destruction	rd
4	barrier de-intensification	bdi
5	railroad repair	rrr
6	road repair	rr
7	barrier mitigation	bm
10	close supporting fire	csf

Nevertheless, the game design data may not support some special activities; IDAHEX warns the player if he has requested an unsupported activity.

The player requests a battle unit to perform a special activity through the activity command.

Initial reply: "activity <name><loc1> <loc2>"

The argument name must be one of the abbreviated special activity names shown above.

The rest of the command sequence depends on whether the activity is LOC modification--types 1 through 7--or close supporting fire.

Case 1. LOC Modification Activity

The arguments loc1 and loc2 are the numbers of the two adjacent, active cells where the activity should occur; that is, the activity will affect the barrier, rail link, or road link between cell loc1 amd loc2. The player completes specification of the activity by identifying the battle unit that should perform it and the fraction of the unit's resources that should actually participate in the activity.

Prompting: Specify unit and level of effort.

Reply: "<unitnbr> <lvl>"

The argument unitnbr must be the identification number of a friendly battle unit; lvl must be a fixed decimal constant, $0 < lvl \leq 1$.

Unit unitnbr will be able to perform the activity when and only when it is located in cell locl or cell loc2 and its side owns its location. The unit will continue performing the activity, or trying to perform it, until the activity is terminated.

Example The player wants battle unit 102 to commit three-fourths of ' s resources to the task of destroying the railroad between the adjacent cells 44 and 68:

```
Enter command.
"activ rrd 44 58"
Specify unit and level of effort.
"102 .75"
```

Example. The player wants unit 102 to commit one-fourth of its resources to the task of intensifying the barrier between cells 44 and 68 (which may mean destroying bridges):

```
Enter command.
"activ bi 44 68"
Specify unit and level of effort.
"102 .25"
```

A unit may perform more than one special activity at the same time, but it may not commit more resources than it has to special activities. IDAHEX refuses a player's request that a unit perform an LOC modification activity if the unit's total level of effort in all LOC modification activities would exceed 1; performing close supporting fire activities does not reduce a unit's capacity for LOC modification activities. LOC modification activities can be terminated through the cancel command.

Case 2. Close Supporting Fire Activity

The argument locl is the number of the cell from which the close supporting fire is to come. Cell locl must be owned by the player, but it need not contain battle units. Nevertheless, no close supporting fire can result from the activity until one of the player's units assumes a hold posture in the cell. The argument loc2 is the number of the cell that is the location of the engagement for which supporting fire is requested. The activity request is accepted even if there is no engagement in cell loc2, but no fire occurs until an engagement begins there.

Finally, the player selects the intensity of supporting fire.

Prompting: Intensity of fire?

Reply: "<lvl>"

The argument <lvl> is the fraction of weapons capable of supporting fire that will fire. (Alternatively, the capable weapons' average intensity of fire.)

The total intensity of fire of the player's close supporting fire activities may not exceed 1. IDAHEX refuses a new request that would violate this constraint. The constraint is independent of LOC modification activities.

Close supporting fire activities can be terminated through the cancel command. A close supporting fire activity is cancelled automatically if the cell from which the fire comes falls to the enemy.

4.6 THE CANCEL COMMAND

The cancel command cancels a mission and dissolves the corresponding task force, cancels a delivery order, or terminates a special activity.

Initial replay: "cancel <cd><number1><number2>"

The argument cd must be the word "mission" (or "miss") to cancel a mission, "delivery" (or "deli") to cancel a delivery order, "locma" to terminate an LOC modification activity, and "fire" to terminate a supporting fire activity. In canceling a mission, number1 is its number, and the argument number2 is not used. In canceling a delivery order, number1 is the identification number of the delivery task force, and the number2 is the cell number of the delivery destination. In terminating an LOC modification activity, number1 and number2 are the numbers of the two adjacent cells where the activity is occurring. In terminating a supporting fire activity, number1 is the cell from which the fire comes, and number2 is the cell to which it goes (the engagement location in the case of close supporting fire).

If an LOC modification activity is being terminated, IDAHEX requests additional information:

Prompting: Enter activity and unit.

Reply: "<abact> <unitnbr>"

The argument abact is the abbreviated name of the special activity:

special activity	abbreviation
barrier intensification	bi
railroad destruction	rrd
road destruction	rd
barrier de-intensification	bdi
railroad repair	rrr
road repair	rr
barrier mitigation	bm

The argument unitnbr is the number of the unit performing the activity. Both arguments may be omitted or, equivalently, made blank or 0:

- If abact = "   " but unitnbr > 0, every special activity of unit unitnbr on the designated LOC (between cell number1 and cell number2) is terminated.
- If unitnbr is omitted but abact is not, every special activity whose abbreviated name is abact being performed on the designated LOC by friendly units is terminated.
- If abact and unitnbr are both omitted, every special activity being performed on the designated LOC by friendly units is terminated.

In terminating a supporting fire activity, either or both of the arguments number1 and number2 may be 0 (or missing):

- If number1 > 0 and number2 = 0, all the player's supporting fire from cell number1 is canceled.
- If number1 = 0 and number2 > 0, all the player's supporting fire directed to cell number2 is canceled.
- If both arguments are 0, all the player's supporting fire is canceled.

Example. The player cancels mission 29:

```
Enter command.
"canc miss 29"
```

Example. The player cancels the delivery order for a delivery by task force 4 to cell 10:

```
Enter command.
"canc deli 4 10"
```

Example. The player terminates all special activities of unit 102 on the LOC between the adjacent cells 44 and 68:

```
Enter command.
"canc locma 44 68"
Enter activity and unit.
",102"
```

4.7 ABANDONING RESOURCES

A player may want to lighten one of his battle units to increase its mobility. The abandon command causes the abandonment and destruction of specified quantities of a battle unit's resources.

Initial reply: "abandon <unitnbr>"

The argument unitnbr is the number of the battle unit that should abandon resources. IDAHEX requests the player to list the quantities to be abandoned.

Prompting: List integral quantities to be abandoned, in order.

Reply: "<q1>, <q2>,..., <qn>"

The item q1 is the quantity of the unit's type 1 resources to be abandoned, q2 is the quantity of type 2 resources to be abandoned, and so on. Each item should be a nonnegative integer. The abandoned resources are immediately deducted from the unit's stocks; they cease to exist.

Example. Suppose the player's side has six types of resources. The player commands the destruction of 10 type 2 resources and 12 type 5 resources in battle unit 4:

```
Enter command.
"abandon 4"
List integral quantities to be
abandoned, in order.
"0 10 0 0 12"
```

A player cannot use the abandon command to destroy enemy resources: IDAHEX prevents a player from issuing commands to enemy units.

4.8 GETTING AIR SUPPORT

The air command involves the IDAHEX entry point air permitting the player to request air strikes. When he is finished requesting air strikes, he may enter other commands. He may give the air command any number of times during his turn.

Initial reply: "air"

4.9 SUSPENDING THE PLAYER TURN

The end command signifies that the player is finished entering commands at this time but may want to enter more before the game proceeds.

Initial reply: "end"

4.10 ENDING THE PLAYER TURN

The go command signifies that the player is finished entering commands and is ready for game action to continue. When both players have issued the go command, the game proceeds.

Initial reply: "go"

4.11 STOPPING THE GAME

The stop command requests IDAHEX to stop the game by exiting from the IDAHEX computer program.

Initial replay: "stop"

Example:

```
Enter command.
"stop"
Do you wish to stop the war?
(All work not saved will be lost.)
yes
```

The word yes is entered by the player in response to IDAHEX's question. Note that it is not enclosed in quotation marks.

Any response other than yes causes the stop command to be ignored.

The game situation can be saved prior to issuing the stop command by issuing the save command.

4.12 DISPLAYING INFORMATION

The display command is a request for information to be written on the player's terminal. The capabilities of IDAHEX's information display facility may exceed those described in this subsection; a player can learn the capabilities of the version he is using by entering the command "help display".

Initial reply: "display <keywd> <number1> <number2>"

The argument keywd is a word indicating the kind of information to be displayed. It is chosen from one of the following: "missions", "deliveries", "status", "unit", "balance", "who", "activities", "trafficability". The following subsections describe the information obtainable under the various values of keywd.

4.12.1 Missions

Initial reply: "display missions <number1> <number2>"

If number1 > 0 and number2 = 0 (or, equivalently, the item number2 is omitted), the command causes display of information on mission number1 (and therefore task force number1) provided the task force belongs to the player's side. If number2 $\geq$ number1 > 0, the command causes display of information on each of the player's missions numbered between number1 and number2. If number1 = number2 = 0, the command causes display of information on all of the player's missions.

Example. The player wants information on the composition and mission of task force 29, which belongs to this side.

```
Enter command.
"disp miss 29"
```

Example. The player wants information on all his task forces and their missions.

```
Enter command.
"disp miss"
```

4.12.2 Delivery Orders

Initial reply: "display deliveries number1 number2 "

If number1 = 0 and number2 = 0, the command causes display of information on all the player's outstanding delivery orders. If number1 > 0, delivery orders whose delivery task force is not task force number1 are excluded. If number2 > 0, delivery orders whose delivery destination is not cell number2 are excluded.

Example. The player requests information on all his delivery orders:

```
Enter command.
"disp deli"
```

Example. The player requests information on all delivery orders designating task force 29 as the delivery task force:

```
Enter command.
"display delivery 29"
```

No information will be displayed unless task force 29 belongs to his side.

Example. The player requests information on all his delivery orders for cell 10:

```
Enter command.
"disp deli,,10"
```

4.12.3 Battle Unit Status

Initial reply: "display status <unitnbr1> <unitnbr2>"

If unitnbr1 > 0 and unitnbr2 = 0, the command causes display of the status of battle unit unitnbr1 provided it is active. If unitnbr2 ≥ unitnbr1 > 0, the command causes display of the status of every active unit numbered between unitnbr1 and unitnbr2. If unitnbr1 = unitnbr2 = 0, the command causes display of the status of every active unit.

Example. The player wishes to learn the status of battle unit 4:

```
Enter command.
"disp stat 4"
```

Example. The player wishes to learn the status of all battle units:

```
Enter command.
"disp stat"
```

4.12.4 Battle Unit Status and Resources

Initial reply: "display unit <unitnbr>"

If unitnbr > 0, the command causes display of information on the status and resources of battle unit unitnbr. If unitnbr = 0, the command causes display of information on the status and resources of every unit, including enemy units.

Example. The player requests information on battle unit 4:

```
Enter command.
"disp unit 4"
```

Example. The player requests information on all battle units:

```
Enter command.
"display units"
```

4.12.5 Battle Unit Resources Available for Combat

IDAHEX groups engaged battle units into combat forces. Two units belong to the same combat force if and only if they are from the same side, participating in the same engagement, and located in the same cell. (A unit can never participate in more than one engagement at the same time.) The members of a combat force share their support resources and use their weapons in concert. The quantity of the force's weapons of a given type that can participate in combat may be reduced if the force fails to have enough supplies and personnel of various types to meet its resources' requirements for support; also, certain types of weapons can not participate in combat without the protection of other weapons. The "fractional involvement" of a given type of equipment is the fraction of the force's equipment of that type that can participate in combat; the fractional involvement of a given type of support resources is the fraction of the force's support resources of that type that are available and required to support other resources.

Initial reply: "display balance <unitnbr>"

If unitnbr > 0, IDAHEX interprets it as the identification number of an engaged battle unit, finds the combat force to which the unit belongs, and displays fractional involvement data for the force. Alternatively, if unitnbr = 0, IDAHEX requests the player to list a group of one or more units, which will be regarded as though it be a combat force in finding the fractional involvement of its resources. (The only restriction is that these units must all belong to the same side.)

Example. Suppose units 11, 13, and 69 comprise the set of Red units participating in a given engagement and located in a given cell. The player wants to learn their resources' fractional involvement.

```
Enter command.
"display balance 13"
```

The preceding command is equivalent to each of the following two commands:

```
"disp bala 11"
"disp bala 69"
```

Example. The player wants to learn what the fractional involvement of the resources in units 58, 60, and 69 would be if these units presently constituted a combat force:

```
Enter command.
"disp bala"
List units.
"60, 58, 69"
```

IDAHEX displays the fractional involvement of each type of resources (in the indicated combat force or the specified group of units) on a line labeled "f.i.". On the next line IDAHEX displays the "delta number" of each type of resources. The delta number of a given type of equipment, say type i, is defined as P - Q; P is the total quantity of type i equipment that could be protected by the weapons of the combat force or group of units, assuming the fractional involvement of every type of weapon were 1; Q is the total quantity of type i equipment in the combat force or group of units. If type i equipment can protect itself, P - Q = +∞, which is written as a string of asterisks. Unprotected equipment can not participate in combat. The delta number of a given type of support resources, say type k, is defined as S - D; D is the aggregate demand for type k support by the resources in the combat force or group of units; S is the total quantity of type k support resources in the combat force or group of units. Thus, a negative delta number for a type of weapons may indicate that their participation in combat is limited by a shortage of weapons capable of protecting them; a negative delta number for a type of support resources may indicate that participation of weapons in combat is limited by a shortage of this type of support.

4.12.6 Battle Units by Location

Initial reply: "display who cellnbr "

The command causes display of the identification number of every active unit in cell cellnbr.

Example. The player wants a list of the active units--enemy as well as friendly--in cell 9:

```
Enter command.
"disp who 9"
```

4.12.7 Special Activities

```
Initial reply:  "display activity <cell1> <cell2>"
Prompting:      Indicate activity type and unit.
Reply:          "<abact> <unitnbr>"
```

The command causes display of every special activity with the designated attributes; if a character-string argument is missing or blank, or if an arithmetic argument is missing or 0, the corresponding attribute is ignored.

Example. The player requests information on every special activity affecting any road, railroad, or barrier between cell 24 and cell 48 (which is adjacent):

```
Enter command.
"disp acti 24 48"
Indicate activity type and unit.
""
```

The reply "disp acti 48 24" is equivalent.

Example. The player requests information on all special activities being performed by unit 86:

```
Enter command.
"disp acti"
Indicate activity type and unit.
",86"
```

Example. The player requests information on all railroad destruction activities affecting rail links emanating from cell 48:

```
Enter command.
"disp acti 48"
Indicate activity type and unit.
"rrd"
```

4.12.8 Trafficability

```
Initial reply:   "display trafficability"
Prompting:       Specify route.
Reply:           "<cell1>, <cell2>,...,<celln>"
```

If the arguments cell1,...,celln are omitted, the command causes display of all significant information concerning trafficability in the area of war, including road damage, rail damage, barrier intensification, and barrier mitigation. If the arguments cell1,...,celln are present, they should be the numbers of a sequence of cells, each of which is adjacent to the next; the command causes display of all significant information concerning the trafficability of the route.

Example. Refer to Figure 2.1. The player requests information on the trafficability of the route from cell 18, to cell 25, to cell 33, to cell 34.

```
Enter command.
"disp traff"
Specify route.
"18 25 33 34"
```

He could have given the cell numbers in reverse order.

4.13 SAVING AND RESTORING GAME SITUATIONS

The save command causes IDAHEX to save a complete description of the present game situation together with all the game design data except *mapter, maprd, maprr,* and *mapbar* in an unformatted file.

```
Initial reply:  "save"
```

IDAHEX requests the player to identify by number the file in which the data should be stored.

Example. The player saves the current situation in file 91:

```
Enter command.
"save"
Enter number (65-99) of file in which
game situation should be saved.
"91"
```

IDAHEX requests a file number greater than 60 to discourage the player from giving the number of a file such as file 50 or file 60 that should not be overwritten.

The restore command causes IDAHEX to read game design data and a complete description of a game situation from an unformatted file created by a previous save.

Initial reply: "restore"

Example. The player restores the game situation, including all missions and delivery orders, exactly as it was at the time of the last save command (by either player) designating file 91:

```
Enter command.
"rest"
Enter number of file from which
game should be restored.
"91"
```

The players are given an opportunity almost immediately after IDAHEX is invoked to restore a game situation that was saved by a save command given in an earlier invocation of IDAHEX. They do this not by giving a restore command but by responding to a specific prompting on the Red player's terminal. This permits IDAHEX to acquire all the game design data except *mapter, maprd, maprr,* and *mapbar* from the unformatted file in which the game situation was saved rather than the formatted file 50; *mapter, maprd, maprr,* and *mapbar* are still acquired from file 60.

Immediately after a game situation is restored. IDAHEX inquires whether certain categories of game design data should be re-read from file 50, the formatted file from which it read the game design data before the start of the game. Since file 50 may have been revised in the meantime, re-reading sections of it provides a way of revising the design data in the middle of a game. The new design data take effect immediately, but have no effect on the restored game situation. (Thus, revising the combat data can affect future engagements, but not past ones.)

4.14 GETTING HELP

In its simplest form, the command is invoked as follows:

```
Enter command.
"help"
```

In response IDAHEX lists on the player's terminal all the commands recognized by the IDAHEX version he is using. IDAHEX also explains how the help command can be used to obtain information on the other commands.

4.15 SUMMARY OF COMMANDS

Initial reply:	"abandon <unit_id>"
Prompting:	List integral quantities to be abandoned, in order.
Reply:	"<quantity1> <quantity2>...<quantityn>"
Initial reply[1]:	"activity <name> <cell_id1> <cell_id2>"
Prompting:	Specify unit and level of effort.
Reply:	"<unit_id> <real_number>"
Initial reply:	"air"
Initial reply:	"attach <unit_id> <task_force_id>"
Initial reply:	"cancel delivery <task_force_id> <cell_id>"
Initial reply:	"cancel mission <task_force_id>"

[1] The argument name may be one of the following: bi, rrd, rd, bdi, rrr, rr, bm.

Initial reply:	"cancel fire <cell-id1><cell-id2>"
Initial reply:	"cancel locma <cell-id1><cell-id2>"
Prompting:	Indicate activity type and unit.
Reply:[1]	"<name> <unit-id>"
Initial reply:	"deliver <task_force_id> <cell_id>"
Prompting:	Enter relative size of delivery.
Reply:	"<real_number>"
Prompting:	List recipients.
Reply:	"<id1> <id2>...<idn>"
Initial reply:	"display activity <cell_id1> <cell_id2>"
Prompting:	Indicate activity type and unit.
Reply:	"<activity_name> <unit_id>"
Initial reply:	"display balance <unit_id>"
Prompting :	List units.
Reply :	"<unit_id1> <unit_id2>...<unit_idn>"
Initial reply:	"display deliveries <task_force_id> <cell_id>"
Initial reply:	"display missions <mission_id1> <mission_id2>"
Initial reply:	"display status <unit_id1> <unit_id2>"
Initial reply:	"display trafficability"
Prompting:	Specify route.
Reply:	"<cell_id1> <cell_id2>...<cell_idn>"
Initial reply:	"display unit <unit_id>"
Initial reply:	"display who <cell_id>"
Initial reply:	"detach <unit_id>"
Initial reply:	"end"
Initial reply:	"go"
Initial reply:	"help <command_name>"

[1] The argument name may be one of the following: bi, rrd, rd, bdi, rrr, rr, bm.

[2] Occurs if and only if unit_id ≤ 0.

[3] *Ibid.*

```
Initial reply[1]:  "mission <mission_id>"
Prompting:         Enter orders.
Replies:           "<objective1> <posture1> <time1>"
                          ⋮            ⋮          ⋮

                   "<objectiven> <posturen> <timen>"
Prompting:         List task force.
Reply:             "<unit_id1> <unit_id2>...<unit_idn>"

Initial reply:     "restore"
Prompting:         Enter number of file from which
                   game should be restored.
Reply:             "<file_number>"

Initial reply:     "save"
Prompting:         Enter number (65-99) of file in which
                   game situation should be saved.
Reply:             "<file_number>"

Initial reply:     "send<cell_id>"
Prompting:         List resource givers.
Reply:             "<id1> <id2>...<idn>"
Prompting:         List resource takers.
Reply[2]:          "<keywd> <id1> <id2>...<idn>"
Prompting[3]:      List units to be excluded.
Reply[3]:          "<unit_id1> <unit_id2>...<unit_idn>"
Prompting:         Enter resource index and amount to be
                   transferred, line by line.
Replies:           "<resource_number1> <amount1>"
                              ⋮              ⋮

                   "<resource_numbern> <amountn>"

Initial reply:     "stop"
```

[1] The argument mission_id should be omitted unless the goal is to modify an existing mission, whose number equals mission_id.

[2] If the argument keywd is present, it must be the word "interval" or the word "subset".

[3] This occurs only if the argument keywd equals "subset".

```
Initial reply:    "transfer <cell_id>"
Prompting:        List resource givers.
Reply:            "<id1> <id2>...<idn>"
Prompting:        List resource takers.
Reply:            "<id1> <id2>...<idn>"
Prompting:        Enter resource index and amount to be
                  transferred, line by line.
Replies:          "<resource_number1> <amount1>"
                              :            :
                  "<resource_numbern> <amountn>"
```

5. COMBAT

An engagement arises when a battle unit attempts to occupy a cell containing enemy units in hold or disengagement postures. To be precise, an engagement arises when a task force in an attack posture attempts to enter a hold posture in its objective while its objective contains one or more enemy units in hold or disengagement postures. Its objective then becomes the "engagement location". The task force constitutes the engagement "attackers". Other units from the task force's side may join the engagement subsequently, possibly attacking from different directions; they, too, become "attackers". The enemy units whose location is the attackers' objective and whose postures are hold or disengagement constitute the engagement "defenders". Thus, at the outset of the engagement, one side is the attacker and the other side is the defender. These roles remain fixed throughout the engagement. That does not mean that defenders cannot counterattack: a successful counterattack precipitates a new engagement in which some units that were defenders become attackers and some that were attackers become defenders.

An engagement ends when all its attackers have left or all its defenders have left. If an attacker's location is not the engagement location, it leaves the engagement when its objective becomes a cell other than the engagement location. If an attacker's location is the engagement location, it leaves the engagement when it enters a posture class other than 1 or 2. A defen er leaves the engagement when it enters a posture class other than 1 or 2.

A defender may leave its engagement by being destroyed, but ordinarily one leaves by entering a movement posture. While it is in a movement posture, the enemy cannot engage it. That is one reason for the disengagement delay, and specifically, for making one term of the delay proportional to the anticipated movement delay. Loosely speaking, if the tactical situation implies that the unit is vulnerable to pursuit by engagement attackers, its disengagement delay (hence, the interval during which it is engaged) is extended to account for the combat that its rearguard might actually have with pursuing units.

Each engagement has a stylized FEBA that measures the attackers' progress. The variable feba expresses the FEBA position as a fraction of the cell depth. At the start of a given engagement, feba = 0. At that point all the attackers are in attack postures oriented toward the engagement location. If the attackers are sufficiently strong relative to the defenders, the FEBA advances--feba increases, to a maximum of 1. One might imagine that when feba is increasing, the attackers are beating back the defenders; a more general, and more contemporary, interpretation is that the attackers are penetrating the defenders' formation. The game design datum *febad* is the criterion for deciding when the attackers have penetrated sufficiently to be allowed to occupy the engagement location. As soon as feba ≥ *febad*, the attackers are allowed to enter hold postures in the engagement location, ownership of it passes to their side, and the defenders must disengage and move out or be destroyed.

An engagement's FEBA is independent of other engagements' FEBAs and the general disposition of forces in the area of war. It may be interpreted as a measure of the attackers' penetration of the engagement location, but essentially it is just an abstraction used to determine how long the engagement lasts before the attackers defeat the defenders.

5.1 ATTRITION

This subsection outlines how losses are determined in a given engagement. Section 6.1 of Volume 2 explains fully.

Only ground-to-ground weapons can destroy enemy resources in ground combat. The game design data fix the basic rate at which each type of Red ground-to-ground weapon can disable (destroy or damage) each type of Blue materiel and the basic rate at which each type of Blue ground-to-ground weapon can disable each type of Red materiel. Ground-to-ground weapons do not disable personnel directly; rather, losses of materiel of a given type caused by an enemy weapon of a given type imply certain personnel losses. The basic rate at which a given type of weapon can disable a given type of materiel must be adjusted for: (1) the weapon's allocation of fire, which depends on the composition of the total enemy force in the engagement; (2) the posture of the battle unit to which the weapon belongs; (3) the posture of the battle unit to which the materiel belongs; (4) the environment in the engagement location. If the weapon belongs to an attacker, further adjustments may be needed for: (5) any combat barrier between the attacker's location and the engagement location; (6) the defense preparation time of the unit to which the materiel belongs. After all these adjustments are made, the result is a "potential kill rate" for every combination of shooting ground-to-ground weapon type, target materiel type, shooting unit, and target unit; define K(i,j,m,n) as the potential rate at which a type i weapon belonging to unit m disables type j materiel belonging to unit n, where unit m and unit n are on opposite sides in the engagement.

Let unit m be a participant in the engagement. Let i be a type of ground-to-ground weapon belonging to it. The quantity of type i weapons in unit m that can participate in combat is limited by two factors: support and protection. First, assuming that the support resources category of the unit's side is nonempty, insufficient quantities of support resources may limit the fraction of type i weapons that can participate in combat. Second, certain types of weapons may not participate in combat unless certain other types are participating in sufficient quantities to protect them. The first factor means that a force cannot employ its weapons unless they are supplied and manned. The second enforces a crude combined arms concept; typically, it means that a force must have enough small arms participating in combat to protect its participating tanks, and enough participating small arms and tanks to protect its participating artillery. The force to which unit m belongs, for the purpose of determining its weapons' participation in combat, is the set of every unit that is from the same side, participating in the same engagement, and located in the same cell as unit m. The units in the force share their support and employ their weapons in concert so that each has the same fraction of weapons of a given type participating in combat.[1]

Once the quantity of participating type i weapons in unit m is found, it is adjusted to get the effective quantity, which may differ because a very low or very high density of friendly forces in the cell where unit m is located may degrade its weapons' effectiveness. The quantity ERS(m,i) is defined as the effective quantity of type i weapons in unit m that can participate in combat.

If unit m is an engagement attacker, let unit n be any defender; if unit m is a defender, let unit n be any attacker. The potential loss rate of type j materiel in unit n due to fire from the type i ground-to-ground weapons in unit m is computed as

$$K(i,j,m,n) * ERS(m,i).$$

[1] The quantity of a unit's type i weapons participating in combat, divided by the quantity it has, defines the "fractional involvement" of its type i resources. This number can be found through the command "display balance".

Let nw be the number of types of ground-to-ground weapons for the side to which unit m belongs. The total potential loss rate of type j materiel in unit n due to all fire from unit m is

$$\sum_{i=1}^{nw} K(i,j,m,n) * ERS(m,i).$$

Summing over all the friends of unit m in the engagement would yield the total potential loss rate of type j materiel in unit n. Two additional adjustment factors are applied to the potential loss rate in the process of finding the actual loss rate.

The first adjustment, which the game designer may choose to omit, scales loss rates according to the loss rates that would be predicted from the engagement's ground force ratio. Because the loss rates computed above obey the Lanchester square law, it is appropriate to determine the values of the weapons in the engagement by the antipotential potential method. (See Section 6.1.2 of Volume 2.) The method, as IDAHEX uses it, generates a value for each type of the attackers' ground-to-ground weapons and each type of the defenders' ground-to-ground weapons. A weapon's value reflects its ability to disable enemy weapons under the specific conditions of the engagement; the value of a given type of Red or Blue weapon usually differs from engagement to engagement and changes over the course of any one engagement. Let v(i) be the value of a type i ground-to-ground weapon held by the attackers in this engagement at this time. Let unit m_1, unit m_2,..., unit m_k be the attackers. Their total value is

$$\sum_{j=1}^{k} \sum_{i=1}^{nw} ERS(m_j,i) * v(i),$$

where nw is the number of types of the attackers' ground-to-ground weapons. The defenders' total value is computed analogously. The ground force ratio is defined as the ratio of the attackers' total value to the defenders'. If the game designer wants losses scaled according to the ground force ratio, the game design data include curves that predict the fraction of their total value that the attackers lose and the fraction of their total value that the defenders lose given the ground force ratio, the attackers' posture and the defenders' posture. The potential loss rates are scaled to agree with this prediction.

The second, and final, adjustment scales the potential loss rate according to the intensity of combat, which is lowest when feba = 0 and highest when feba indicates that the attackers have penetrated to the depth of the defense.

Unless the ground force ratio is almost 0 or is extremely large, a unit's actual loss of a given type of materiel is assessed as the lesser of its adjusted potential loss and the quantity of its materiel of that type participating in combat. The portion of the actual loss attributable to each type of enemy weapon is computed, and then the unit's losses of personnel are inferred.

Depending on the type of enemy weapon responsible for disabling it, lost equipment is classified as irreparable, severely damaged, or moderately damaged. Damaged but reparable equipment is assigned to repair pools provided the game designer chose to play maintenance. When equipment is repaired, it is returned to the battle unit that lost it.

5.2 FEBA MOVEMENT

As noted earlier in this section, in a given engagement the variable feba measures the attackers' progress. Its value always lies between 0 and 1. Its rate of change at any moment of the engagement depends upon a force ratio that includes the contribution of close air support (CAS). Air support is assessed periodically, and losses of ground-to-ground weapons inflicted by CAS are recorded for use by the ground combat procedure. To find a force ratio that reflects both the ground forces and the air forces in the engagement, it is necessary to value them consistently. The antipotential potential method offers a way of valuing the attackers' close air support that is completely consistent with the way the attackers' total value is determined (Section 5.1), and it offers a way of valuing the defenders' close air support that is completely consistent with the way the defenders' total value is determined. The combined ground-air force ratio is defined as

$$\frac{A1 + A2}{D1 + D2} ,$$

where A1 is the attackers' total (ground) value, A2 is the value of their CAS, D1 is the defender's total (ground) value, and D2 is the value of their CAS. The rate of change of the variable feba depends upon the combined ground-air force ratio, the attackers' posture, the defenders' posture, and the side to which the attackers belong. This rate may be negative.

There is a game design datum called *febad*; $0 < \textit{febad} \leq 1$. If, at some point in the engagement, feba becomes greater than or equal to *febad*, the defenders are declared defeated and required to retreat--to disengage and move to an adjacent cell. IDAHEX judges whether they have a line of retreat and, if so, selects the adjacent cell toward which they should go. If they have no line of retreat, they are destroyed. The procedure for finding and selecting a line of retreat is explained in Section 6.3 of Volume 2.

6. AIR SUPPORT

A player initiates input of air strikes by giving the command "air". IDAHEX includes no air warfare model and therefore has no way of ascertaining what air assets a side can allocate against enemy ground forces. It assumes that any air strike a player enters is within his side's capability. In practice the game designer adopts either of two solutions: he gives each player a list of the air assets available at various times in the game for use against enemy ground forces, or he runs an air warfare model concurrently with IDAHEX.

An air strike is specified in a sequence of requests by IDAHEX and replies by the player. The player's replies are interpreted according to the rules described at the beginning of Section 4.

IDAHEX requests input of an air strike by requesting the player to "Enter strike role and target."

Reply: "<abname> <cellnbr1> <cellnbr2>"

If the argument abname is the word "end", it indicates that the player does not wish to enter any more air strikes at the moment.[1] Alternatively, abname must be one of the words in the left-hand column of the following list of strike roles:

bi	barrier intensification
rrd	railroad damage
rd	road damage
das	deep air support
cas	close air support

The word "CAS" is equivalent to "cas". Barrier intensification typically represents an attack on bridges. The first three strike roles all imply an effort to degrade the line of communication between two adjacent cells. Close air support (CAS) is an attack on enemy battle units participating in a specific engagement with friendly battle units. Deep air support is any other attack on enemy battle units.

[1] His player turn does not end. He could enter more air strikes after issuing the air command again.

If the strike role is CAS, cellnbr1 is the number of the cell where the support is desired (the engagement location), and the argument cellnbr2 is superfluous. Usually, there is only one engagement in progress at a given cell. Should there be more than one engagement at cell cellnbr1, IDAHEX asks the player to resolve the ambiguity by naming one of the units, enemy or friendly, participating in the engagement in which he wants close air support. If the strike role is deep air support, cellnbr1 is the number of the cell containing the enemy units to be attacked, and the argument cellnbr2 is superfluous. If the strike role is barrier intensification, railroad damage, or road damage, the strike affects the barrier, rail link, or road link between cell cellnbr1 and cell cellnbr2, which must be adjacent.

If the strike role is deep air support, the player must assign priorities to attacks on units in various posture classes:

Prompting: Enter target posture classes in order of priority.

Reply: "<asprty(1)> <asprty(2)> <asprty(3)> <asprty(4)>"

The reply must contain four items--the integers 1, 2, 3, and 4, in any order. The highest priority posture class is asprty(1), and the lowest is asprty(4).

Next, IDAHEX asks for the composition of the strike:

Prompting: List, in order, number of each type of aircraft in strike.

Reply: "<q1> <q2>...<qn>"

Each item in the reply must be a nonnegative integer. The i-th item is the number of type i aircraft participating in the strike.[1]

The game design data fix for each side the number of types of aircraft; the number of types of air-to-ground weapons; and the notional load of each type of aircraft--the quantities of the various types of air-to-ground weapons that it carries. Knowing the strike's composition, IDAHEX can ascertain the total quantity of each type of air-to-ground weapon delivered.

[1] The game designer must give the players a list showing the various types of aircraft and their type numbers.

Example. The player inputs an air strike in support of his units engaged in cell 124:

```
Enter strike role and target.
"cas 124"
List, in order, number of each type
of aircraft in strike.
"25 25 0 13"
```

Example. The player wishes to attack moving enemy units whose location is cell 45:

```
Enter strike role and target.
"das 45"
Enter target posture classes in order
of priority.
"3 2 1 4"
List, in order, number of each type
of aircraft in strike.
"0 15"
```

Example. The player wishes to cut the rail lines between cell 46 and cell 70 (which are adjacent):

```
Enter strike role and target.
"rrd 46 70"
List, in order, number of each type
of aircraft in strike.
"0 0 13 0 2"
```

Example. The player terminates his input of air strikes for this cycle.

```
Enter strike role and target.
"end"
```

Section 8 explains how air-to-ground weapons delivered in an air strike can produce barrier intensification, railroad damage, or road damage. The strike roles that imply damage to enemy battle units are close air support and deep air support To assess the damage resulting from a CAS or deep air support strike, IDAHEX constructs a set V, the set of battle units that are victims of the strike.

Suppose the strike role is CAS. If there is no engagement whose location is the target cell, the player is warned and no strike occurs. If such an engagement exists, let V be the set of every enemy unit in the engagement. If the enemy units are the engagement defenders and at least one of them is in a hold posture, then delete from V every unit in a disengagement posture.

On the other hand, suppose the strike role is deep air support. Let k be the smallest positive integer such that asprty(k) equals the posture class of some enemy unit located in the target cell. Thus, asprty(k) is the highest priority posture class that appears among enemy units in the target cell. Let V be the set of every active enemy unit whose location is the target cell and whose posture class is asprty(k). This definition of V implicitly assumes that the strike aircraft can only distinguish enemy units from each other by location (cell) or posture class.

Choose any unit in V. The quantity of the unit's materiel of a given type that is destroyed in the strike depends upon the composition of the strike; the allocation of the air-to-ground weapons' fire, which depends on the resource composition of V; the unit's posture; and the environment in the target cell. Destruction of a given type of materiel by a given type of air-to-ground weapon induces losses of the various types of personnel.

7. SUPPLIES CONSUMPTION

Only an active unit consumes supplies. To be precise, the unit's resources consume supplies. Consumption of supplies of a given type by resources of a given type depends on the unit's posture and, if it is engaged, the fraction of the resources that are participating in combat. One qualification must be added: passenger resources (resources being transported by other resources) in a moving unit consume supplies as though in a hold posture. A unit that does not belong to a task force consumes supplies out of its own stocks. The supplies in a task force are pooled to meet the consumption of all its elements, and every element's stock of a given type of supplies is reduced by the same factor.

A unit's stocks of supplies cannot become negative. If the stock of a given type of supplies in a unit or task force falls to 0, the unit or task force simply does not consume any more. It suffers no immediate penalty, such as spontaneous attrition, as a result. Nevertheless, lacking supplies of some type might prevent it from moving (might make its movement delay infinite) and, should it become engaged, might prevent some or possibly all of its weapons from participating in combat.

8. DAMAGE, REPAIR, AND IMPROVEMENT OF LINES OF COMMUNICATION

Ground resources and air-to-ground weapons may render sections of rail links unusable by, for example, tearing up or deforming the tracks. Such activity is termed *railroad destruction*. Ground resources and air-to-ground weapons may render sections of road links unusable by, for example, destroying the roadbed or laying mines. Such activity is termed *road destruction*. Ground resources may repair damage to rail or road links. Ground resources and air-to-ground weapons may make barriers harder to cross: they may, for example, destroy bridges over rivers or defiles, precipitate landslides in mountain passes, and lay mine belts. Such activity is termed *barrier intensification*. Ground resources may counter barrier intensification efforts: they may, for example, repair bridges, clear landslides, and clear mine belts. Such activity is termed *barrier de-intensification*. Ground resources may make barriers easier to cross: they may, for example, build bridges over rivers or defiles, smash walls, and tunnel through ridges. Such activity is termed *barrier mitigation*. Road destruction, railroad destruction, barrier intensification, road repair, railroad repair, barrier de-intensification, and barrier mitigation collectively are termed LOC (line of communication) modification activities.

Recall from Section 3.2 that resources trying to move by rail suffer an infinite movement delay if the rail link between their location and their objective is damaged, and resources trying to move entirely by road or by some combination of road and cross-country movement must move cross-country to the extent that the road link is damaged. A player can learn whether a road or rail link between two adjacent cells is damaged, and to what extent, through the command "display trafficability."

Whereas road or rail destruction makes sections of a road or rail link unusable without altering the road or rail type, barrier intensification's only possible effect is to transform the barrier into a different type. For example, barrier intensification might transform a barrier representing a bridged river into a barrier representing an unbridged river; in this example the barrier intensification represents destruction of bridges. Barrier de-intensification transforms a barrier that has suffered barrier intensification back to its original type. For example, a bridged river might be transformed by barrier intensification into an unbridged river, and subsequently restored by barrier de-intensification to a bridged river. Barrier mitigation transforms a barrier into a less formidable barrier. For example, it might transform an unbridged river into a bridged river. If a barrier subjected to barrier intensification is later subjected to barrier mitigation, the mitigation behaves exactly as de-intensification until no vestiges of intensification remain: continuing the previous examples, if the bridges over a river are made unusable by barrier intensification, barrier mitigation causes rebuilding of the old bridges before building of new ones.

Battle units can be requested to perform LOC modification activities by using the activity command. Note that the command requests a specific battle unit to perform a specific type of LOC modification on a road link, rail link, or barrier between two adjacent active cells. The player may issue the request at any time, but the battle unit will be unable to comply until its location is one of the designated cells and its side owns the cell. IDAHEX may cancel an LOC modification activity when nothing remains to be done; for example, it cancels a railroad repair activity when no damage to the link remains.

A battle unit may consume additional supplies while it performs an LOC modification activity.

Certain types of LOC modification--road destruction, railroad destruction, and barrier intensification--can be accomplished by air strikes. Section 6 explains how to request such a strike.